Malicious URL Detection based on Machine Learning

Cho Do Xuan[1], Hoa Dinh Nguyen[1]
Information Security dept, Posts and Telecommunications Institute of Technology, Hanoi, Vietnam[1, 2]
Information Assurance dept, FPT University, Hanoi, Vietnam[1]

Tisenko Victor Nikolaevich[3]
Systems of automatic Design
Peter the Great St. Petersburg Polytechnic University
Russia, St.Petersburg
Polytechnicheskaya, 29

***Abstract*—Currently, the risk of network information insecurity is increasing rapidly in number and level of danger. The methods mostly used by hackers today is to attack end-to-end technology and exploit human vulnerabilities. These techniques include social engineering, phishing, pharming, etc. One of the steps in conducting these attacks is to deceive users with malicious Uniform Resource Locators (URLs). As a results, malicious URL detection is of great interest nowadays. There have been several scientific studies showing a number of methods to detect malicious URLs based on machine learning and deep learning techniques. In this paper, we propose a malicious URL detection method using machine learning techniques based on our proposed URL behaviors and attributes. Moreover, bigdata technology is also exploited to improve the capability of detection malicious URLs based on abnormal behaviors. In short, the proposed detection system consists of a new set of URLs features and behaviors, a machine learning algorithm, and a bigdata technology. The experimental results show that the proposed URL attributes and behavior can help improve the ability to detect malicious URL significantly. This is suggested that the proposed system may be considered as an optimized and friendly used solution for malicious URL detection.**

Keywords—URL; malicious URL detection; feature extraction; feature selection; machine learning

I. INTRODUCTION

Uniform Resource Locator (URL) is used to refer to resources on the Internet. In [1], Sahoo et al. presented about the characteristics and two basic components of the URL as: protocol identifier, which indicates what protocol to use, and resource name, which specifies the IP address or the domain name where the resource is located. It can be seen that each URL has a specific structure and format. Attackers often try to change one or more components of the URL's structure to deceive users for spreading their malicious URL. Malicious URLs are known as links that adversely affect users. These URLs will redirect users to resources or pages on which attackers can execute codes on users' computers, redirect users to unwanted sites, malicious website, or other phishing site, or malware download. Malicious URLs can also be hidden in download links that are deemed safe and can spread quickly through file and message sharing in shared networks. Some attack techniques that use malicious URLs include [2, 3, 4]: Drive-by Download, Phishing and Social Engineering, and Spam.

According to statistics presented in [5], in 2019, the attacks using spreading malicious URL technique are ranked first among the 10 most common attack techniques. Especially, according to this statistic, the three main URL spreading techniques, which are malicious URLs, botnet URLs, and phishing URLs, increase in number of attacks as well as danger level.

From the statistics of the increase in the number of malicious URL distributions over the consecutive years, it is clear that there is a need to study and apply techniques or methods to detect and prevent these malicious URLs.

Regarding the problem of detecting malicious URLs, there are two main trends at present as malicious URL detection based on signs or sets of rules, and malicious URL detection based on behavior analysis techniques [1, 2]. The method of detecting malicious URLs based on a set of markers or rules can quickly and accurately detect malicious URLs. However, this method is not capable of detecting new malicious URLs that are not in the set of predefined signs or rules. The method of detecting malicious URLs based on behavior analysis techniques adopt machine learning or deep learning algorithms to classify URLs based on their behaviors. In this paper, machine learning algorithms are utilized to classify URLs based on their attributes. The paper also includes a new URL attribute extraction method.

In our research, machine learning algorithms are used to classify URLs based on the features and behaviors of URLs. The features are extracted from static and dynamic behaviors of URLs and are new to the literature. Those newly proposed features are the main contribution of the research. Machine learning algorithms are a part of the whole malicious URL detection system. Two supervised machine learning algorithms are used, Support vector machine (SVM) and Random forest (RF).

The paper is organized as follows. Section II reviews some recent works in the literature on malicious URL detection. The proposed malicious URLs detection system using machine learning is presented in Section III. In this section, the new features for URLs detection process are also described in details. Experimental results and discussions are provided in Section IV. The paper is concluded by Section V.

II. RELATED WORKS

A. Signature based Malicious URL Detection

Studies on malicious URL detection using the signature sets had been investigated and applied long time ago [6, 7, 8]. Most of these studies often use lists of known malicious URLs. Whenever a new URL is accessed, a database query is

executed. If the URL is blacklisted, it is considered as malicious, and then, a warning will be generated; otherwise URLs will be considered as safe. The main disadvantage of this approach is that it will be very difficult to detect new malicious URLs that are not in the given list.

B. Machine Learning based Malicious URL Detection

There are three types of machine learning algorithms that can be applied on malicious URL detection methods, including supervised learning, unsupervised learning, and semi-supervised learning. And the detection methods are based on URL behaviors.

In [1], a number of malicious URL systems based on machine learning algorithms have been investigated. Those machine learing algorithms include SVM, Logistic Regression, Nave Bayes, Decision Trees, Ensembles, Online Learning, ect. In this paper, the two algorithms, RF and SVM, are used. The accuracy of these two algorithms with different parameters setups will be presented in the experimental results.

The behaviors and characteristics of URLs can be divided into two main groups, static and dynamic. In their studies [9, 10, 11] authors presented methods of analyzing and extracting static behavior of URLs, including Lexical, Content, Host, and Popularity-based. The machine learning algorithms used in these studies are Online Learning algorithms and SVM. Malicious URL detection using dynamic actions of URLs is presented in [12, 13]. In this paper, URL attributes are extracted based on both static and dynamic behaviors. Some attribute groups are investigated, including Character and semantic groups; Abnormal group in websites and Host-based group; Correlated group.

C. Malicious URL Detection Tools

- URL Void: URL Void is a URL checking program using multiple engines and blacklists of domains. Some examples of URL Void are Google SafeBrowsing, Norton SafeWeb and MyWOT. The advantage of the Void URL tool is its compatibility with many different browsers as well as it can support many other testing services. The main disadvantage of the Void URL tool is that the malicious URL detection process relies heavily on a given set of signatures.
- UnMask Parasites: Unmask Parasites is a URL testing tool by downloading provided links, parsing Hypertext Markup Language (HTML) codes, especially external links, iframes and JavaScript. The advantage of this tool is that it can detect iframe fast and accurately. However, this tool is only useful if the user has suspected something strange happening on their sites.
- Dr.Web Anti-Virus Link Checker: Dr.Web Anti-Virus Link Checker is an add-on for Chrome, Firefox, Opera, and IE to automatically find and scan malicious content on a download link on all social networking links such as Facebook, Vk.com, Google+.
- Comodo Site Inspector: This is a malware and security hole detection tool. This helps users check URLs or enables webmasters to set up daily checks by downloading all the specified sites. and run them in a sandbox browser environment.
- Some other tools: Among aforementioned typical tools, there are some other URL checking tools, such as UnShorten.it, VirusTotal, Norton Safe Web, SiteAdvisor (by McAfee), Sucuri, Browser Defender, Online Link Scan, and Google Safe Browsing Diagnostic.

From the analysis and evaluation of malicious URL detection tools presented above, it is found that the majority of current malicious URL detection tools are signature-based URL detection systems. Therefore, the effectiveness of these tools is limited.

III. Malicious URL Detecting using Machine Learning

A. The Model

Fig. 1 presents the proposed malicious URL detection system using machine learning. The malicious URL detection model using machine learning contains two stages: training and detection.

- Training stage: To detect malicious URLs, it is necessary to collect both malicious URLs and clean URLs. Then, all the malicious and clean URLs are correctly labeled and proceeded to attribute extraction. These attributes will be the best basis for determining which URLs are clean and which are malicious. Details of these attributes will be presented in details in this paper. Finally, this dataset is divided into 2 subsets: training data used for training machine learning algorithms, and testing data used for testing process. If the classification performance of the machine learning model is good (high classification accuracy), the model will be used in the detection phase.
- Detection phase: The detection phase is performed on each input URL. First, the URL will go through attribute extraction process. Next, these attributes are input to the classifier to classify whether the URL is clean or malicious.

B. URL Attribute Extraction and Selection

In [1], the authors listed some main attribute groups for malicious URL detection as follows.

Lexical features: these features include URL length, main domain length, maximum token domain length, path average length, average token length in domain.

Host-based Features: these features are extracted from the host characteristics of the URLs. These attributes indicate the location of malicious servers, the identity of malicious servers, the degree of impact of several host-based features that contribute the URL's malicious level.

Content-based Features: these features are acquired when a whole web page is downloaded. The workload of these features is quite heavy, since a lot of information needs to be extracted, and there may be security concerns about accessing that URL. However, with more information available about a particular

site, it is expected to create a better prediction model. The content-based features of a website can be extracted primarily from its HTML content and the use of JavaScript.

Above are the three main attribute groups commonly used by researchers to detect malicious URLs. However, each study has its own decision on suitable attributes and characteristics for each particular experimental dataset. In this paper, the use of all three attribute groups is recommended. However, in each attribute group some new attributes and characteristics of the URL to optimize the ability to detect malicious URLs are proposed. The new attributes for malicious URL detection in this research are listed in Tables I, II, and III.

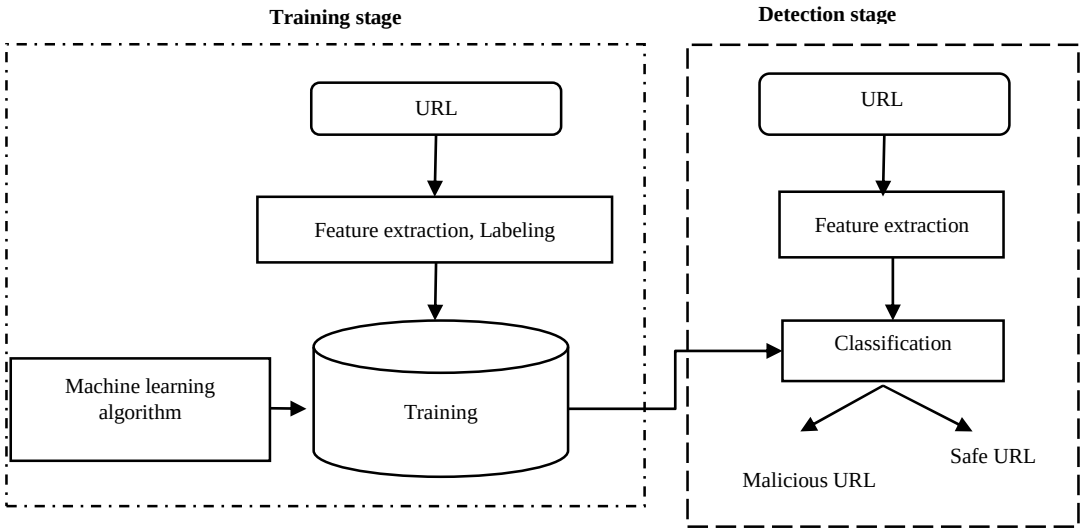

Fig. 1. Malicious URL Detection Model using Machine Learning.

TABLE. I. LIST OF URL FEATURES IN LEXICAL FEATURE GROUP

No	Feature group	Feature	Data type	Description
1	Lexical group	NumDots	numeric	Number of character '.' in URL
2		SubdomainLevel	numeric	Number of subdomain levels
3		PathLevel	numeric	The depth of URL
4		UrlLength	numeric	The length of URL
5		NumDash	numeric	Number of the dash character '-'
6		NumDashInHostname	numeric	Number of dash character in the hostname
7		AtSymbol	boolean	There exists a character '@' in URL
8		TildeSymbol	boolean	There exists a character '~' in URL
9		NumUnderscore	numeric	Number of the underscore character
10		NumPercent	numeric	Number of the character '%'
11		NumQueryComponents	numeric	Number of the query components
12		NumAmpersand	numeric	Number of the character '&'
13		NumHash	numeric	Number of the character '#'
14		NumNumericChars	numeric	Number of the numeric character
15		NoHttps	boolean	Check if there exists a HTTPS in website URL
16		IpAddress	boolean	Check if the IP address is used in the hostname of the website URL
17		DomainInSubdomains	boolean	Check if TLD or ccTLD is used as a part of the subdomain in website URL
18		DomainInPaths	boolean	Check if TLD or ccTLD is used in the link of website URL
19		HttpsInHostname	boolean	Check if HTTPS is disordered in the hostname of website URL
20		HostnameLength	numeric	Length of hostname
21		PathLength	numeric	Length of the link path
22		QueryLength	numeric	Length of the query
23		DoubleSlashInPath	boolean	There exists a slash '//' in the link path
24		NumSensitiveWords	numeric	Number of sensitive words (i.e., “secure”, “account”, “webscr”, “login”, “ebayisapi”, “sign in”, “banking”, “confirm”) in website
25		EmbeddedBrandName	boolean	There exists a brand name in the domain
26		PctExtHyperlinks*	float	The percentage of external hyper links in the HTML source code of website

TABLE. II. LIST OF URL FEATURE IN THE HOST-BASED FEATURE GROUP

No	Feature group	Feature	Data type	Description
27	Host-based feature group	PctExtResourceUrls*	float	Percentage of URL external resource in HTML source codes of website
28		ExtFavicon*	boolean	Check if favicon is installed from a hostname different from the URL hostname of website
29		InsecureForms*	boolean	Check if actions in the form containing the contend of URL without HTTPS protocol
30		RelativeFormAction*	boolean	Check if the action form contains a relative URL
31		ExtFormAction*	boolean	Check if the action form contains an external URL
32		AbnormalFormAction*	boolean	Check if the action form contains an abnormal URL.
33		PctNullSelfRedirectHyperlinks*	float	Percentage of hyperlinks containing an empty value, an auto-redirecting value, such as "#", URL of current website, or some abnormal values such as "file://E:/"
34		FrequentDomainNameMismatch	boolean	Check if the most frequent hostname in the HTML source code does not match the URL of website.
35		FakeLinkInStatusBar*	boolean	Check if HTML source code contains a JavaScript command on MouseOver to display a fake URL in the status bar
36		RightClickDisabled	boolean	Check if HTML source code contains a JavaScript command to turn off the right click of the mouse
37		PopUpWindow	boolean	Check if HTML source code contains a JavaScript command to start a popup window
38		SubmitInfoToEmail	boolean	Check if HTML source code contains "mailto" in the HTML
39		IframeOrFrame	boolean	Check if iframe or frame is used in HTML source codes
40		MissingTitle	boolean	Check if the title tag is empty in HTML source codes
41		src_eval_cnt	int	Number of function eval () in HTML source codes
42		src_escape_cnt	int	Number of function escape () in HTML source codes
43		src_exec_cnt	int	Number of function exec() in HTML source codes
44		src_search_cnt	int	Number of function search() HTML source codes
45		ImagesOnlyInForm*	boolean	Check if actions in the form of HTML source code does not contain text, but only images
46		rank_country	Boolean	Current country rank of website URL is in top 1 million of Alexa
47		rank_host	Boolean	The rank of the host website URL is in top 1 million of Alexa
48		AgeDomain	int	The age of domain since it is registered

TABLE. III. LIST OF URL FEATURES IN CORRELATED FEATURE GROUP

No	Feature group	Feature	Data type	Description
49	correlated feature group	UrlLengthRT*	-1, 0, 1	Correlated length of URL
50		PctExtResourceUrlsRT*	-1, 0, 1	Correlated percentage of external URL
51		AbnormalExtFormActionR*	-1, 0, 1	Correlated abnormal actions in form
52		ExtMetaScriptLinkRT*	-1, 0, 1	Correlated meta script link
53		SubdomainLevelRT*	-1, 0, 1	Correlated sub-domain level
54		PctExtNullSelfRedirectHyperlinksRT *	-1, 0, 1	Correlated null self-redirect hyperlinks

All attributes marked "*" in Tables I, II, III are newly extracted and selected in this research. Besides, in previous researches, authors tend to use feature extraction and selection method based on a group of predefined features. However, those recommended features are specialized and not popular. As a results, it is usually difficult to implement those features in other works, and to re-evaluate the detection performance of those features. In this work, we try to combine basic features to formulate new ones.

C. Machine Learning Algorithm Selection

The application of machine learning algorithms in detecting malicious URLs has been studied and applied widely [1]. In this paper, two commonly used supervised machine learning algorithms, RF and SVM [14, 15], are used.

In this research, machine learning algorithms are the last puzzle to complete our proposed malicious URL detection system. Those algorithms are suitable to utilized the usefulness of our new features selected for malicious URL detection. The machine learning algorithms are already well investigated in the literature. In this work, SVM and RF are selected as an example to illustrate the good performance of the whole detection system, and are not our main focus. Readers are encouraged to implement some other algorithms such as Naïve Bayes, Decision trees, k-nearest neighbors, neural networks, etc.

In order to explore the effectiveness of using these two algorithms, different adjustments of parameters are implemented.

IV. Experimental Results

A. Dataset and Experiment Environments

1) Experiment dataset: The experimental dataset for malicious URL detection model includes: 470.000 URLs collected from [16, 17, 18, 19], of which about 70.000 URLs are malicious and 400.000 URLs are safe. All these URLs are checked by Virus Total tool to verify the labels of each URL. The complete dataset is stored using CSV format. Each URL sample has a label "bad" for malicious and "good" for safe. Details of the data are as follows:

- Phishtank [16]: Phishtank is a service Website dedicated for sharing phishing URLs. Suspicious URLs can be sent to Phishtank for verification. The data in Phishtank is updated hourly.
- URLhaus [17]: URLhaus is a project from abuse.ch aiming at sharing malicious URLs being used for malicious software distribution.
- Alexa [18]: Is a database ranking all websites according to their usefulness.
- Malicious_n_Non-Malicious URL [19]: is a data source with more than 400,000 labeled URL. In this database, 82% of all URLs are safe, while remaining 18% of URLs are malicious.

2) Experimental setup: The dataset of both safe and malicious URLs mentioned above is divided into 2 subsets. About 80% of the dataset, 470.000 URLs (400.000 safe URLs, 70.000 malicious URL), is used for training, and about 20% of the dataset, about 10.000 URLs (5.000 malicious URLs, 5.000 safe URLs), is used for testing. The experiment is repeated many times with both SVM and RF algorithm. Different parameter settings are used in different runs.

3) Experiment dataset

- Setup environment: Python version 3.6; Spark version 2.3.0; Hadoop version 2.7; Java (JDK) 8; Ubuntu 18.04.
- Hardware: RAM 16GB; Intel(R) Xeon(R) CPU E5-2640 v3 @ 2.60GHz.

B. Results and Discussions

1) Evaluation metrics: Accuracy: the percentage of correct decisions among all testing samples

$$acc = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% \quad (1)$$

where: *TP*- True positive is the number of malicious URLs correctly labeled; *FN* - False negative is the number of malicious URLs misclassified as safe; *TN*- True negative is the number of safe URL correctly labeled; *FP* - False positive is the number of safe URLs misclassified as malicious.

Confusion matrix: is a two-way Table IV representing how many samples are classified into which label accordingly.

Precision: is the percentage of malicious URLs correctly labeled (TP) among all malicious URLs labeled by the classifier (TP+FP).

$$precision = \frac{TP}{TP + FP} \times 100\% \quad (2)$$

Recall: is the percentage of malicious URLs correctly labeled (*TP*) among all malicious URLs of the testing data (*TP+FN*).

$$\mathrm{Re}call = \frac{TP}{TP + FN} \times 100\% \quad (3)$$

F1-score: is the harmonic mean of precision and recall. High F1 value means the classifier is good.

$$F1 = \frac{2 \times precision \times \mathrm{Re}call}{precision + \mathrm{Re}call} \quad (4)$$

FPR (False prediction rate) is calculated as:

$$FRP = \frac{FP}{FP + TN} \times 100\% \quad (5)$$

2) Results

- Training performance

To evaluate the training performance of the machine learning algorithm, both two data subsets are used individually. Each of these data subsets has different data size as well as different distribution of data labels, which may result in different training performances. The results are presented in Table V.

Experimental results show that the RF with 100 trees gives the best predictive result. In return, the training time of the RF is slightly longer than SVM, but the testing time is not much different. The accuracy of the second dataset is reduced due to the unbalance between safe and malicious URLs of the data. As expected, RF algorithm, with its fast speed and high accuracy, is very suitable for classification problem. Besides, in our research, when machine learning algorithms are combined with spark libraries, the training and testing time can be reduced significantly. SparkML Machine Learning is a library package that provides and supports many machine learning algorithms such as SVM, RF, Naïve Bayes, Regression, Clustering, Collaborative Filtering, ... It is a suitable tool for applying machine learning algorithms with fast and accurate processing speed on large datasets.

- Testing results: In this paper, additional small testing dataset, with 107 safe URLs and 118 malicious URLs, is used to evaluate the performance of the best machine learning algorithm discussed above, RF (100). The results are presented in Table VI.

Confusion matrix parameters: TP: 92.174%; FPR: 12.037%; TN: 87.963%; FN: 7.826%

TABLE. IV. Confusion Matrix

	Classified malicious URL	**Classified safe URL**
Real malicious URLs	TP	FN
Real safe URLs	FP	TN

TABLE. V. TRAINING PERFORMANCE OF MALICIOUS URL DETECTION SYSTEM

Dataset	Algorithm and parameters	Accuracy (%)	Precision (%)	Recall (%)	Training time (s)	Testing time (s)
10.000 URLs	SVM (100 iterations)	93.39	94.67	92.51	2.32	0.01
	SVM (10 iterations)	93.35	94.84	92.71	3.11	0.01
	RF (10 trees)	99.10	98.43	97.45	2.78	0.01
	RF (100 trees)	99.77	98.75	97.85	3.34	0.01
470.000 URLs	SVM (100 iterations)	90.70	93.43	88.45	272.97	2.12
	SVM (10 iterations)	91.07	93.75	88.85	280.33	2.31
	RF (10 trees)	95.45	90.21	95.12	372.97	2.02
	RF (100 trees)	96.28	91.44	94.42	480.33	2.30

TABLE. VI. TESTING RESULTS

	Predicted safe URL	Predicted malicious URL
Real safe URL (107)	96	11
Real malicious URL (118)	9	109

V. CONCLUSIONS

In this paper, a method for malicious URL detection using machine learning is presented. The empirical results in Tables V and VI have shown the effectiveness of the proposed extracted attributes. In this study, we do not use special attributes, nor do we seek to create huge datasets to improve the accuracy of the system as many other traditional publications. Here, the combination between easy-to-calculate attributes and big data processing technologies to ensure the balance of the two factors is the processing time and accuracy of the system. The results of this research can be applied and implemented in information security technologies in information security systems. The results of this article have been used to build a free tool [20] to detect malicious URLs on web browsers.

REFERENCES

[1] D. Sahoo, C. Liu, S.C.H. Hoi, "Malicious URL Detection using Machine Learning: A Survey". CoRR, abs/1701.07179, 2017.

[2] M. Khonji, Y. Iraqi, and A. Jones, "Phishing detection: a literature survey," IEEE Communications Surveys & Tutorials, vol. 15, no. 4, pp. 2091–2121, 2013.

[3] M. Cova, C. Kruegel, and G. Vigna, "Detection and analysis of drive-by-download attacks and malicious javascript code," in Proceedings of the 19th international conference on World wide web. ACM, 2010, pp. 281–290.

[4] R. Heartfield and G. Loukas, "A taxonomy of attacks and a survey of defence mechanisms for semantic social engineering attacks," ACM Computing Surveys (CSUR), vol. 48, no. 3, p. 37, 2015.

[5] Internet Security Threat Report (ISTR) 2019–Symantec. https://www.symantec.com/content/dam/symantec/docs/reports/istr-24-2019-en.pdf [Last accessed 10/2019].

[6] S. Sheng, B. Wardman, G. Warner, L. F. Cranor, J. Hong, and C. Zhang, "An empirical analysis of phishing blacklists," in Proceedings of Sixth Conference on Email and Anti-Spam (CEAS), 2009.

[7] C. Seifert, I. Welch, and P. Komisarczuk, "Identification of malicious web pages with static heuristics," in Telecommunication Networks and Applications Conference, 2008. ATNAC 2008. Australasian. IEEE, 2008, pp. 91–96.

[8] S. Sinha, M. Bailey, and F. Jahanian, "Shades of grey: On the effectiveness of reputation-based "blacklists"," in Malicious and Unwanted Software, 2008. MALWARE 2008. 3rd International Conference on. IEEE, 2008, pp. 57–64.

[9] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Identifying suspicious urls: an application of large-scale online learning," in Proceedings of the 26th Annual International Conference on Machine Learning. ACM, 2009, pp. 681–688.

[10] B. Eshete, A. Villafiorita, and K. Weldemariam, "Binspect: Holistic analysis and detection of malicious web pages," in Security and Privacy in Communication Networks. Springer, 2013, pp. 149–166.

[11] S. Purkait, "Phishing counter measures and their effectiveness– literature review," Information Management & Computer Security, vol. 20, no. 5, pp. 382–420, 2012.

[12] Y. Tao, "Suspicious url and device detection by log mining," Ph.D. dissertation, Applied Sciences: School of Computing Science, 2014.

[13] G. Canfora, E. Medvet, F. Mercaldo, and C. A. Visaggio, "Detection of malicious web pages using system calls sequences," in Availability, Reliability, and Security in Information Systems. Springer, 2014, pp. 226–238.

[14] Leo Breiman.: Random Forests. Machine Learning 45 (1), pp. 5- 32, (2001).

[15] Thomas G. Dietterich. Ensemble Methods in Machine Learning. International Workshop on Multiple Classifier Systems, pp 1-15, Cagliari, Italy, 2000.

[16] Developer Information. https://www.phishtank.com/developer_info.php. [Last accessed 11/2019].

[17] URLhaus Database Dump. https://urlhaus.abuse.ch/downloads/csv/. [Ngày truy nhập 11/2019].

[18] Dataset URL. http://downloads.majestic.com/majestic_million.csv. [Last accessed 10/2019].

[19] Malicious_n_Non-MaliciousURL. https://www.kaggle.com/antonyj453/urldataset#data.csv. [Last accessed 11/2019].

[20] chrome.zip. https://drive.google.com/file/d/13G_Ndr4hMFx_qWyTEjHuOyJmHFWD0Gud/view?fbclid=IwAR0SLVCrvjHHGmoHZH97nXN3Bm-DMY7jG4SOsKZYLAZjTFgeoJADfli64-g. [Last accessed 12/2019].